AMENDMENTS TO THE CLAIMS

Please cancel claims 2 and 3, without prejudice or disclaimer of subject matter, and amend claims 1, 4 to 7, 9, 11 to 15, 18 to 23, 25, 26 and 29, as shown below. This listing of claims replaces all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A frame handler for application-level memory management, the frame handler comprising:

an associated block of memory divided into a plurality of frames, with each of the plurality of frames operable to store an indexing structure associated with a single attribute of a data record, and each of the plurality of frames divided into a plurality of instances, with each of the plurality of instances operable to store an index node of the indexing structure ~~such that data elements may be stored in the instances~~;

a data structure identifying the unused instances within each of the plurality of frames ~~the block of memory~~; and

an application interface operable to receive a request for an unused instance from a software application,

wherein the frame handler is operable to identify an unused instance in response to a request received by the application interface.

2. (Cancelled)

3. (Cancelled)

4. (Currently Amended) The frame handler of ~~claim 2~~ claim 1 wherein the data structure ~~includes~~ further comprises a tree structure .

5. (Currently Amended) The frame handler of claim 4 wherein the tree structure is an AVL tree.

6. (Currently Amended) The frame handler of claim 4 wherein the tree structure includes a frame node associated with each ~~frame~~ of the plurality of frames.

7. (Currently Amended) The frame handler of claim 6 wherein each frame node is associated with a list of unused instances within the associated frame.

8. (Original) The frame handler of claim 7 wherein the list of unused instances is represented as a ring structure.

9. (Currently Amended) The frame handler of claim 6 further comprising an anchor including:

an empty list storing each frame node having no unused instances; and

a non-empty list storing each frame node having unused instances.

10. (Original) The frame handler of claim 1 further comprising an operating system interface operable to allocate a block of memory such that the frame handler is operable to allocate an additional block of memory when the block of memory is exhausted.

11. (Currently Amended) A method for allocating memory in a computer system, the method comprising:

outputting a request from an application to an operating system for allocation of a block of memory by the operating system to the application;

accessing the block of memory for the application;

dividing the block of memory into a plurality of frames, with each of the plurality of frames operable to store an indexing structure associated with a single attribute of a data record;

dividing each of the plurality of frames into a plurality of instances, with each of the plurality of instances ~~instance~~ operable to store an index node of the indexing structure ~~data and associated with an application-defined instance type~~; and

maintaining a data structure identifying the unused instances within each of the plurality of frames ~~indicating each unused instance~~.

12. (Currently Amended) The method of claim 11 wherein maintaining a data structure ~~indicating each unused instance includes~~ identifying the unused instances within each of the plurality of frames further comprises creating a frame node corresponding to each of the frames.

13. (Currently Amended) The method of claim 12 wherein maintaining a data structure identifying the unused instances within each of the plurality of frames further comprises ~~indicating each unused instance further includes~~ associating a list of unused instances with each frame node.

14. (Currently Amended) The method of claim 13 wherein associating a list of unused instances with each frame node includes creating a ring data structure comprised of unused instances.

15. (Currently Amended) The method of claim 12 wherein maintaining a data structure identifying the unused instances further comprises ~~indicating each unused instance further includes~~ organizing the frame nodes in a tree structure.

16. (Original) The method of claim 15 wherein the tree structure is an AVL tree.

17. (Original) The method of claim 12 further comprising creating an anchor data structure including a ring including an empty list and a non-empty list.

18. (Currently Amended) The method of claim 17 wherein maintaining a data structure identifying the unused instances further comprises ~~indicating each unused instance further~~

~~includes~~ placing frame nodes with unused instances in the non-empty list and placing nodes without unused instances in the empty list.

19. (Currently Amended) The method of claim 12 wherein dividing the block of memory into the plurality of frames includes associating a frame identifier with each of the plurality of frames.

20. (Currently Amended) The method of claim 19 wherein each frame node includes the frame identifier of its associated frame.

21. (Currently Amended) A method comprising:

accessing a block of memory for an application;

dividing the block of memory into a plurality of frames, including first and second frames, with each of the plurality of frames operable to store an indexing structure associated with a single attribute of a data record;

dividing each of the plurality of frames into a plurality of instances, including first and second lists of instances, with each of the plurality of instances operable to store an index node of the indexing structure;

assigning a first identifier that is associated with ~~a first memory portion~~ the first frame to a first frame node;

linking ~~a first~~ the first list of instances to the first frame node~~, the first list of instances corresponding to divisions of the first memory portion~~;

assigning a second identifier that is associated with ~~a second memory portion~~ the second frame to a second frame node;

linking ~~a second~~ the second list of instances to the second frame node~~, the second list of instances corresponding to divisions of the second memory portion~~;

constructing a data structure using a plurality of nodes including the first node and the second node; and

selecting available instances within the plurality of frames using the data structure, via ~~from the instances for data storage by~~ an application~~, wherein the instances are associated with an application-determined instance type~~.

22. (Currently Amended) The method of claim 21 wherein constructing a data structure further comprises constructing an AVL tree using the plurality of frame nodes.

23. (Currently Amended) The method of claim 22 wherein selecting available instances using the data structure further comprises traversing the data structure to locate the available instances.

24. (Original) The method of claim 22 further comprising superposing a linear list over the data structure, wherein the linear list includes a first pointer to an empty subset of the plurality of nodes that has no associated memory available for use by the application and a second pointer to a not_empty subset that has associated memory available for use by the application.

25. (Currently Amended) The method of claim 24 wherein the first frame node is a first not_empty frame node in the not_empty subset, and wherein selecting available instances further comprises:

following the second pointer to the first frame node; and

using the first list of instances as the available instances.

26. (Currently Amended) The method of claim 25 further comprising:

re-setting the second pointer to a second not_empty frame node in the not_empty subset, and

including the first node in the empty subset.

27. (Original) The method of claim 21 further comprising:

determining an origin list from which the available instances were selected; and

returning the available instances to the origin list.

28. (Original) The method of claim 27 wherein determining the origin list comprises matching an identifier of the available instances to the first identifier or the second identifier.

29. (Currently Amended) The method of claim 28 wherein matching the identifier comprises following a pointer to a first not_empty frame node of a not_empty subset of the plurality of nodes, the not_empty subset including not_empty frame nodes with associated memory available for use by the application.

30. (Original) The method of claim 21 wherein the first memory portion includes a frame into which a block of memory allocated from the operating system is divided.